LA-UR 91-2436

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

LA-UR--91-2436

DE91 016301

TITLE: DEFENSE IN DEPTH AND RESOURCE OPTIMIZATION FOR SAFEGUARDS

AUTHOR(S): A. Zardecki and J. T. Markin

SUBMITTED TO: 32nd Annual Meeting of the Institute of Nuclear Materials Management, New Orleans, Louisiana, July 28-31, 1991



Los Alamos

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

# DEFENSE IN DEPTH AND RESOURCE OPTIMIZATION FOR SAFEGUARDS*

Andrew Zardecki and Jack T. Markin
Safeguards Systems Group, N4
Los Alamos National Laboratory
Los Alamos, NM 87545

## ABSTRACT

The resource allocation problem for safeguards is solved by using dynamic programming. The existing program RAOPS (Resource Allocation Optimization Program for Safeguards) is extended to include both divergent and convergent configurations of activities. The new algorithm is applicable to any configuration that can be described in terms of a tree data structure. To treat the problem of defense in depth, a stochastic optimization—in which the optimization applies to expected values—is utilized. Numerical examples illustrating the general theory are given.

## 1. INTRODUCTION

Selecting the safeguards elements to constitute a system for protecting special nuclear materials (SNM) is a complex process involving choices about those technologies and procedures that are most effective in countering a range of threats. The computer program RAOPS (Resource Allocation Optimization Program for Safeguards) developed at Los Alamos by the Safeguards Systems Group[1] helps the analyst design a safeguards system for a new facility or an upgrade of an existing facility. For a serial arrangement of activities, the program determines the configuration of safeguards options that maximizes the detection probability against a range of scenarios for theft or diversion of SNM under the constraint of fixed safeguards resources. Here, the term "activity" refers to an area or boundary in which one or more safeguards elements can be deployed.

---

*Work supported by the US Department of Energy, Office of Safeguards and Security.

The serial arrangement corresponds to the situation in which the adversary sequentially encounters each safeguards activity. This has earlier been studied by Markin et al.[2] and by Fishbone.[3] Figure 1 shows a typical arrangement, with the corresponding tree structure depicted in Fig. 2.

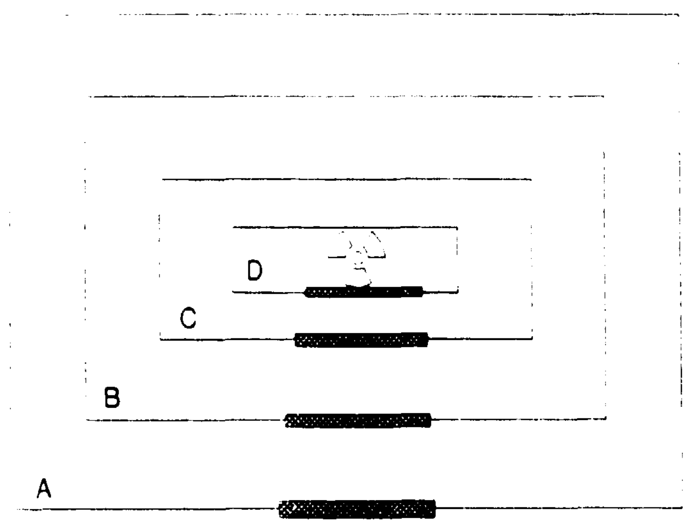


**Fig. 1. Serial configuration of four activities.**


**Fig. 2. The tree (list) structure corresponding to Fig. 1.**

In practice, though, a realistic arrangement will involve multiple choices; for example, after penetrating the main facility gate *A*, followed by the inner wall *B*, the adversary has to select one of the two different storage areas, as shown in Figs. 3 and 4. Figures 3 and 4 may be contrasted with Figs. 1 and 2, respectively. Stated differently, a nonserial configuration implies a structure in which there are branch points in the adversary's path to his goal.

The goal of this paper is to generalize the RAOPS algorithm to include both divergent and convergent configurations of activities, in which the safeguards options operate under risk. In those circumstances, the function

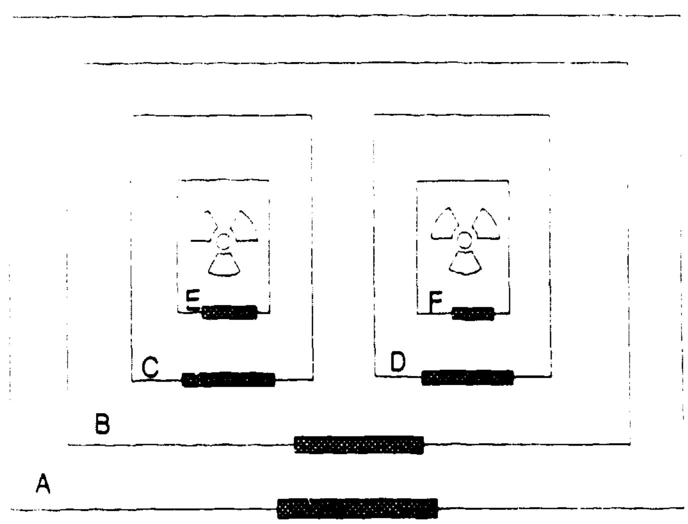


**Fig. 3. Divergent configuration of six activities.**

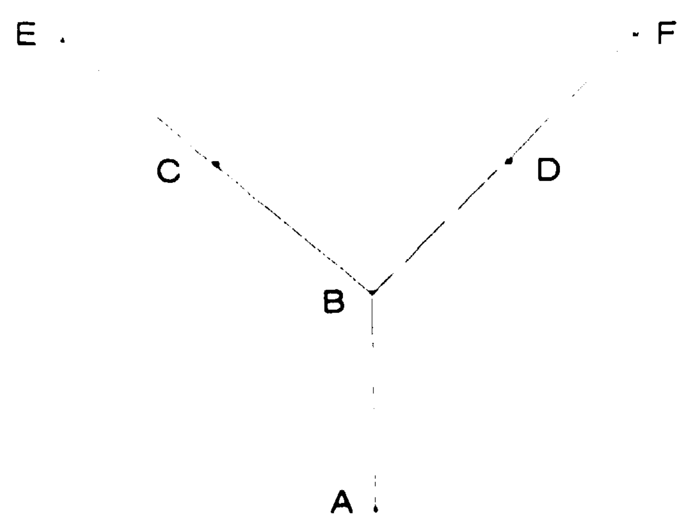


**Fig. 4. The tree structure of activities corresponding to Fig. 3.**

depends on random elements that simulate the option reliability, in addition to the state and decision variables. With prescribed probabilities of success or failure, we can model the system performance under risk, thus addressing an important issue of defense in depth. In this way, we evaluate the performance and localize the system's weakest link.

A word about the terminology is in order here. In the context of safeguards, when we speak of maximizing the detection probability, we mean maximizing the return (objective) function. Stated equivalently, when minimizing the nondetection probability, we minimize the return function.

## 2. DISCRETE DYNAMIC PROGRAMMING ALGORITHM

We start by listing some basic results and by explaining our notation. For a serial system, the multiplicative objective function is

$$J = \prod_{k=1}^{N} L[x(k), u(k), k] \ . \tag{1}$$

Here $x(k)$ and $u(k)$ refer to state variables and decision variables at stage $k$. The system equations describe how the state variables at stage $k + 1$ are related to the state variables at stage $k$. These equations are written as

$$x(k+1) = g[x(k), u(k), k] \ , \tag{2}$$

where $g$ is a known function.

For the resource allocation problem, $g$ is simply a difference of $x$ and $u$:

$$x(k+1) = x(k) - u(k) \ , \tag{3}$$

and the function $L[x(k),u(k),k]$ specifies the nondetection probability at stage k.

The dynamic programming optimization solves the following iterative functional equation for the optimum return $I(x,k)$

$$I(x,k) = \min_{u \in U} \left\{ L(x,u,k) \times I[g(x,u,k), k+1] \right\} \tag{4}$$

for $k = 1, \ldots, N-1$, by minimizing the expression in the curly brackets over the set $U$ of decisions. The starting condition is

$$I(x,k) = \min_{u \in U} \{L(x,u,N)\} \ . \tag{5}$$

This recursion procedure, called backward recursion, solves the initial state problem, in which the optimal $N$-stage return becomes a function of the input state of stage one. When state inversion is possible, as in the resource allocation problem, one can also use forward recursion to solve a final state problem. In the final state problem, the optimal return is found as a function of the stage output. Finally, the initial-final state optimization consists in finding the optimal return as a function of the input to stage one and the output from stage $N$.

This basic dynamic programming procedure prevents the combinatorial explosion (curse of dimensionality) from occurring.[4] The generalization of the basic procedure to nonserial systems will depend on the specific form of the configuration.[5]

## 3. DEFENSE IN DEPTH: STOCHASTIC OPTIMIZATION

In the preceding section, the cost function at a given stage was defined as a deterministic function of the state and decision variables. To accommodate a potential failure of safeguards options, we now introduce—at each stage—a binary stochastic variable $\xi(k)$ controlling the return. The objective function thus becomes

$$J = \prod_{k=1}^{N} L\left[x(k),u(k),\xi(k),k\right] \ , \tag{6}$$

where $\xi(k) = 1$ and 0 refer to success and failure, respectively. We further assume that the probability of success $p(\xi,u,k)$ of a given option $u$ at stage $k$ is known; the probability of failure is then regarded as the probability of a complementary event, equal to $1 - p$.

The solution to this problem of stochastic optimization under risk is again given in terms of a recursion relation, which applies to an expected value of the objective function. The basic equation replacing Eq. (4) is

$$I\left(x,\xi,N\right)=\min_{u\in U}\ \sum_{\xi}\ p\ (\xi,u,k) \tag{7}$$

$$\times\left\{L\ (x,u,\xi,k)\times I\left[g\ (x,u,k),\xi,k+1\right]\right\}\ ,$$

with the starting condition given as

$$I\left(x,\xi,N\right)=\min_{u\in U}\ \sum_{\xi}\ p\ (\xi,u,k)\left\{L\ (x,u,\xi,N)\right\}\ . \tag{8}$$

In Eqs. (7) and (8), the summation over $\xi$ is in fact a weighted sum over the two possibilities of success and failure.

## 4. NUMERICAL RESULTS

The foregoing considerations refer to a serial system, in which the output of a given stage is identical to the input of the stage directly following it. For a divergent configuration, we use the backward recursion to solve the initial state problem for an individual branch of the tree. At branch points, we optimally split the resources among the emanating branches. This is achieved on the basis of the minimax principle, relevant to the worst-case scenario. In other words, we minimize the maximum value that the cost function can take among the branches.

To illustrate our previous considerations, we analyze a tree composed of ten nodes, as illustrated in Fig. 5. Incidentally, Aho et al.[6] use the same topology to elucidate various ways of traversing the tree.

Without loss of generality, we may assume that, for each activity, the matrix of detection probabilities has already been reduced to the detection probability vector. For simplicity, we also neglect the possibility of option combinations. Table I lists, then, the detection probabilities for various costs.

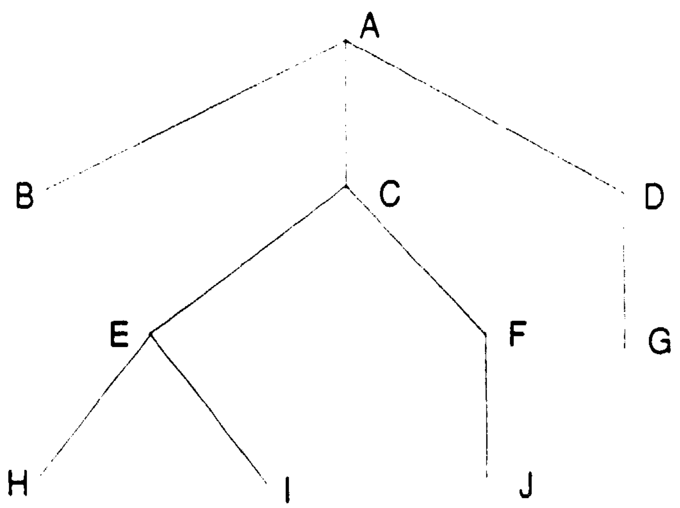


**Fig. 5. Tree of activities.**

**Table I. The detection probabilities for different activities as functions of cost**

| Activity | Option Cost 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A | 0.10 | 0.15 | 0.20 | 0.25 |
| B | 0.25 | 0.30 | 0.45 | 0.50 |
| C | 0.15 | 0.30 | 0.40 | 0.50 |
| D | 0.10 | 0.20 | 0.30 | 0.40 |
| E | 0.10 | 0.20 | | |
| F | 0.15 | 0.30 | | |
| G | 0.10 | 0.20 | 0.25 | |
| H | 0.10 | 0.20 | 0.40 | 0.60 |
| I | 0.30 | | | |
| J | 0.40 | | | |

In Table I, the cost is given in terms of arbitrary units, and the empty entries indicate the lack of an option. In the absence of randomness, assuming the budget of 10 units, we obtain the detection probability of 0.46. The optimal allocation of resources is 1, 3, and 3 units assigned to nodes A, B, and C, respectively. Because the nodes B and C absorb most of the resources, we now investigate how the malfunction of node B affects the allocation problem. To this end, we let fail each option associated with node B, resulting in a probability of detection of 0.1 for each option. We again assume that the probability of success will be 0.5. With the total budget equal again to 10, the detection probability becomes 0.4, whereas the resource assignment is 4, 2, and 2 units at nodes A, B, and C, respectively.

As the results of our example show, there is a strong bias to allocate most of the resources to the nodes neighboring the root of the activities tree. This is due to the minimax optimization at the branch points. Only when the detection probabilities of activities near the root are small, will the trend arise to allocate resources to terminal nodes. We also note that, as formulated, the optimum resource allocation problem is not symmetric with respect to the direction in which the resources flow. For a convergent configuration, we first optimize the individual branches as functions of their inputs and the resources flowing to the branch point. This is different from the divergent configuration, where at each stage we use the backward recursion.

## 5. CONCLUSIONS

We have generalized the optimum resource allocation to include an arbitrary configuration of activities that can be described as a tree data structure. From the standpoint of safeguards, the divergent configuration corresponds to the fixed total budget for a facility; this budget branches to different subfacilities. Although the convergent configuration presents a greater computational challenge, it will probably be used infrequently. It essentially corresponds to a situation in which each subfacility has an independent budget.

To treat the problem of defense in depth, we use the stochastic optimization applied to an expected value of the objective function. This allows us to model the performance of safeguards options. A realistic approach supplements the probabilities of detection with the probability of success, or failure, of the option under consideration.

## REFERENCES


1. J. T. Markin, "Resource Allocation Optimization Program for Safeguards," Los Alamos National Laboratory, Safeguards Systems Group unpublished report N-4/89-422 (1989).
2. J. T. Markin, C. A. Coulter, R. G. Gutmacher, and W. J. Whitty, "Optimizing the Design of International Safeguards Inspection Systems," in *Proceedings Fifth Annual Symposium of Safeguards and Nuclear Material Management* (European Safeguards Research and Development Association, Joint Research Centre, Ispra, Italy, 1983), ESARDA 16, pp. 51-56.
3. L. G. Fishbone, "Optimal Allocation of Safeguards Inspection Resources for a Nuclear Fuel Cycle," Brookhaven National Laboratory report BNL-34127 (December 1973).
4. R. E. Larson and J. L. Casti, *Principles of Dynamic Programming, Parts I and II* (Marcel Dekker, New York and Basel, 1982).
5. R. Aris, G. L. Nemhauser, and D. J. Wilde, "Optimization of Multistage and Branching Systems by Serial Procedure," A. I. Ch. E. Journ. **10**, 913-919 (1964).
6. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms* (Addison Wesley, Reading, Massachusetts, 1987), Sec. 3.1.